09/647300

Speech Recognition and Signal Analysis by straight Search of Subsequences with Maximal Confidence Measure

1 Field of the invention

The invention relates to a common component of:

- Speech Recognition
- Keyword Spotting
- Segments Alignment for DNA and proteins (Human Genome)
- Recognition of Objects in Images

2 Background Art

This invention addresses the problem of *keyword spotting (KWS)* in unconstrained speech without explicit modeling of non-keyword segments (typically done by using filler HMM models or an ergodic HMM composed of context dependent or independent phone models without lexical constraints). Although several algorithms (sometimes referred to as "sliding model methods") tackling this type of problem have already been proposed in the past, e.g., by using Dynamic Time Warping (DTW) [4] or Viterbi matching [9] allowing relaxation of the (begin and endpoint) constraints, these are known to require the use of an "appropriate" normalization of the matching scores since segments of different lengths have then to be compared. However, given this normalization and the relaxation of begin/endpoints, straightforward Dynamic Programming (DP) is no longer optimal (or, in other words, the DP optimality principle is no longer valid) and has to be adapted, involving more memory and CPU. Indeed, at any possible ending time e, the match score of the best warp and start time b of the reference has to be computed [4] (for all possible start times b associated with unpruned paths). Moreover, in [9], and in the same spirit than what is presented here, for all possible ending times e, the average observation likelihood along the most likely state sequence is used as scoring criterion. Finally, this adapted DP quickly becomes even more complex (or intractable) for more advanced scoring criteria (such as the confidence measures mentioned below).

More recently, work in the field of confidence level, and in the framework of hybrid HMM/ANN systems, it was shown [1] that the use of accumulated local posterior probabilities (as obtained at the output of a multilayer perceptron) normalized by the length of the word segment (or, better, involving a double normalization over the number of phones and the number of acoustic frames in each phone) was yielding good confidence measures and good scores for the re-estimation of N-best hypotheses. Similar work, where this kind of confidence measure was compared to several alternative approaches, was reported in [8] and confirmed this conclusion. However, so far, the evaluation of such confidence measures involved the estimation and rescoring of N-best hypotheses. Similar work and conclusions (also using N-best rescoring) were also reported in using likelihood ratio rescoring and non-keyword rejection [7].

2.1 KWS without filler models

Let $X = \{x_1, x_2, \ldots, x_n, \ldots, x_N\}$ denote the sequence of acoustic vectors in which we want to detect a keyword, and let M be the HMM model of a keyword M and consisting of L states $\mathcal{Q} = \{q_1, q_2, \ldots, q_\ell, \ldots, q_L\}$. Assuming that M is matched to a subsequence $X_b^e = \{x_b, \ldots, x_e\}$ $(1 \leq b \leq e \leq N)$ of X, and that we have an implicit (not modeled) *garbage/filler state* q_G preceding and following M, we define (approximate) the log posterior of a model M given a subsequence X_b^e as the average posterior probability along the optimal path, i.e.:

$$\begin{aligned} -\log P(M|X_b^e) &\simeq \frac{1}{e-b+1} \min_{\forall Q \in M} -\log P(Q|X_b^e) \\ &\simeq \frac{1}{e-b+1} \min_{\forall Q \in M} \{-\log P(q^b|q_G) \\ &\quad - \sum_{n=b}^{e-1} [\log P(q^n|x_n) + \log P(q^{n+1}|q^n)] \\ &\quad - \log P(q^e|x_e) - \log P(q_G|q^e)\} \end{aligned} \qquad (1)$$

where $Q = \{q^b, q^{b+1}, \ldots, q^e\}$ represents one of the possible paths of length $(e-b+1)$ in M, and q^n the HMM state visited at time n along Q, with $q^n \in \mathcal{Q}$. In this expression, q_G represents the "garbage" (filler) state which is simply used here as the non-emitting initial and final state of M. Transition probabilities $P(q^b|q_G)$ and $P(q_G|q^e)$ can be interpreted as the keyword entrance and exit penalties, as optimized in [3], but these have not been optimized here. In our case, local posteriors $P(q_\ell|x_n)$ were estimated as output values of a multilayer perceptron (MLP) used in a hybrid HMM/ANN system [2].

For a specific sub-sequence X_b^e, expression (1) can easily be estimated by dynamic programming since the sub-sequence and the associated normalizing factor $(e-b+1)$ are given. However, in the case of keyword spotting, this expression should be estimated for all possible begin/endpoint pairs $\{b,e\}$ (as well as for all possible word models), and we define the matching score of X on M as:

$$S(M|X) = -\log P(M|X_{b^*}^{e^*}) \tag{2}$$

where the optimal begin/endpoints $\{b^*, e^*\}$, and the associated optimal path Q^*, are the ones yielding the lowest average local posterior:

$$\langle Q^*, b^*, e^* \rangle = \underset{\{Q,b,e\}}{\operatorname{argmin}} \frac{-1}{e-b+1} \log P(Q|X_b^e) \tag{3}$$

Of course, in the case of several keywords, all possible models will have to be evaluated.

As shown in [1, 8], a double averaging involving the number of frames per phone and the number of phones will usually yield slightly better performance:

$$\langle Q^*, b^*, e^* \rangle = \tag{4}$$
$$\underset{\{Q,b,e\}}{\operatorname{argmin}} \frac{-1}{J} \sum_{j=1}^{J} \left(\frac{1}{e_j - b_j + 1} \sum_{n=b_j}^{e_j} \log P(q_j^n|x_n) \right)$$

where J represents the number of phones in the hypothesized keyword model and q_j^n the hypothesized phone q_j for input frame x_n.

However, given the time normalization and the relaxation of begin/endpoints, straightforward DP is no longer optimal and has to be adapted, usually involving more memory and CPU. A new (and simple) solution to this problem is proposed in Section 3.1.

2.2 Filler-based KWS

Although various solutions have been proposed towards the direct optimization of (2) as, e.g., in [4, 9], most of the keyword spotting approaches today prefer to preserve the optimality and simplicity of Viterbi DP by modeling the complete input [5] and explicitly [6] or implicitly [3] modeling non-keyword segments by using so called filler or garbage models as additional reference models. In this case, we assume that non-keyword segments are modeled by extraneous garbage models/states q_G (and grammatical constraints ruling the possible keyword/non-keyword sequences).

Let us consider only the case of detecting one keyword per utterance at a time. In this case, the keyword spotting problem amounts at matching the whole sequence X of length N onto an extended HMM model $\overline{M}$ consisting of the states $\{q_G, q_1, \ldots, q_L, q_G\}$, in which a path (of length N) is denoted $\overline{Q} = \{\overbrace{q_G, \ldots q_G}^{b-1}, q^b, q^{b+1}, \ldots, q^e, \overbrace{q_G, \ldots q_G}^{N-e}\}$ with $(b-1)$ garbage states q_G preceding q^b and $(N-e)$ states q_G following q^e, and respectively emitting the vector sequences X_1^{b-1} and X_{e+1}^N associated with the non-keyword segments.

Given some estimation of $P(q_G|x_n)$ (e.g., using probability density functions trained on non keyword utterances), the optimal path $\overline{Q^*}$ (and, consequently b^* and e^*) is then given by:

$$\begin{aligned}
\overline{Q^*} &= \underset{\forall \overline{Q} \in \overline{M}}{\operatorname{argmin}} - \log P(\overline{Q}|X) \\
&= \underset{\forall \overline{Q} \in \overline{M}}{\operatorname{argmin}} \{- \log P(Q|X_b^e) \\
&- \sum_{n=1}^{b-1} \log P(q_G|x_n) - \sum_{n=e+1}^{N} \log P(q_G|x_n)\}
\end{aligned} \qquad (5)$$

which can be solved by straightforward DP (since all paths have the same length). The main problem of filler-based keyword spotting approaches is then to find ways to best estimate $P(q_G|x_n)$ in order to minimize the error introduced by the approximations. In [3], this value was defined as the average of the N best local scores while, in other approaches, this value is generated from explicit filler HMMs. However, these approaches will usually not lead to the "optimal" solution given by (2).

3 Disclosure of Invention

3.1 Iterating Viterbi Decoding (IVD)

In the following, we show that it is possible to define an iterative process, referred to as *Iterating Viterbi Decoding (IVD)* with good/fast convergence properties, estimating the value of $P(q_G|x_n)$ such that straightforward DP (5) yields exactly the same segmentation (and recognition results) than (3). While the same result could be achieved through a modified DP in which all possible combinations (all possible begin/endpoints) would be taken into account, it is possible to show that the algorithm proposed below is more efficient (in terms of both CPU and memory requirements).

Here, I will use a similar scoring technique for keyword spotting without explicit filler model. Compared to previously devised "sliding model" methods (such as [4, 9]), the first algorithm proposed here is based on:

1. A matching score defined as the average observation posterior along the most likely state sequence. It is indeed believed that local posteriors (or likelihood ratios, as in [7]) are more appropriate to the task.

2. The iteration of a Viterbi decoding algorithm, which does not require scoring for all begin/endpoints or N-best rescoring, and which can be proved to (quickly) converge to the "optimal" (from the point of view of the chosen scoring functions) solution without requiring any specific filler models, using straightforward Viterbi alignments (similar to regular filler-based KWS, but at the cost of a few iterations).

3.2 IVD: Description

The IVD algorithm is based on the same criterion than the filler based approaches (5), but rather than looking for explicit (and empirical) estimates of $P(q_G|x_n)$ we aim at mathematically estimating its value (which will be different and adapted to each utterance) such that solving (5) is equivalent to solving (3). Thus, we perform an iterative estimation of $P(q_G|x_n)$, such that the segmentation resulting of (5) is the same than what would be obtained from (3).

Defining $\varepsilon = -\log P(q_G|x_n)$, the proposed algorithm can be summarized as follows:

1. Start from an initial value $\varepsilon_0 = \varepsilon$ (it is actually proven that the iterative process presented here will always converge to the same solution (in more or less cycles, with the worst case upper bound of N iterations) independently of this initialization), (e.g., with ε equal with a cheap estimation of the score of a "match"). In the experiments reported below, ε was initialized to $-\log$ of the maximum of the local probabilities $P(q_k|x_n)$ for each frame x_n.

   An alternative choice could be to initialize ε_0 to a pre-defined score that expression (1) should reach to declare a keyword "matching" (see point 4 below). In this last case, if ε increases at the first iteration, then we can (as proven) directly infer that the match will be rejected, otherwise it will be accepted.

2. Given the current estimate ε_t of $P(q_G|x_n)$ at iteration t, find the optimal path $\langle \overline{Q}_t, b_t, e_t \rangle$ according to (5) and matching the complete input.

3. Update ($t = t+1$) the estimated value of ε_t, defined as the average of the local posteriors along the optimal path Q_t (matching the $X_{b_t}^{e_t}$ resulting of (5) on the keyword model) i.e.:

$$\varepsilon_{t+1} = -\frac{1}{(e_t - b_t + 1)} \log P(Q_t|X_{b_t}^{e_t}) \tag{6}$$

4. Return to (2) and iterate until convergence. If we are not interested in the optimal segmentation, this process could also be stopped as soon as ε reaches a (pre-defined) minimum threshold below which we can declare that a keyword has been detected.

Correctness and convergence proof of this process and generalization to other criteria, are available: each IVD iteration (from the second iteration) will decrease the value of ε_t, and the final path yields the same solution than (3).

3.3 One-pass keyword spotting

3.3.1 General Description

The above algorithm has a very good experimental convergence speed (3-5 iterations in our tests). However, the worst case theoretical convergence speed of the process is N. For this reason, a one step computation is potentially interesting. In the next subsection we show that the standard DP cannot be used for solving the equation (3).

3.3.2 The Principle of Optimality

Let us define $T(\overline{M}, X)$ as the DP table of emission probabilities for an utterance X and the states of the hypothesized word W. When solving by standard DP, we would compute for each entry of the table $T(\overline{M}, X)$ at frame k of X and state s of $\overline{M}$ three values: S_{ks}, L_{ks} and C_{ks}, where S_{ks} corresponds to the sum of the posteriors on the optimal path that leads to the entry, L_{ks} holds the length of the optimal path computed so far, and C_{ks} is the estimation of the cost on the optimal expanded path.

By a path leading to an entry $T(k, s)$ we mean a sequence of entries in the table T, such that there is exactly an entry for each time frame $t \leq k$. At each entry $T(k, s)$, DP selects a locally optimal path noted P_{ks}.

At each step k, we consider all pairs of entries of table $T(\overline{M}, X)$ of type $T(k,s), T(k-1,t)$. We update for each such pair, the current cost C_{ks} (initially ∞), by comparing it with the alternative given by:

$$S_{ks} = S_{(k-1)t} - \log p(s|x_k)p(s|t)$$

$$L_{ks} = L_{(k-1)t} + 1, \forall t > 0, t \leq L$$

$$C_{ks} = \frac{S_k}{L_k} \tag{7}$$

wanting to have at step k the path P_{ks} from the paths $P_{(k-1)t}$ that minimizes C_{NL}. With DP, one will choose the P_{ks} with minimal C_{ks}.

In order for the previous computation to be correct, the optimality principle needs to be respected. The optimality principle of Dynamic Programming requires that the path to the frame $k-1$ that minimizes C_{NL}, also minimizes C_{ks} for an entry at frame k of table $T(\overline{M}, X)$. We have proved that the expression 7 does not respect the optimality principle of Dynamic Programming

3.3.3 Pruning with beam search

The Dynamic Programming can be viewed as a set of safe prunings that are applied at each entry of the DP table and has the property that only one alternative is maintained. We have thus shown that Dynamic Programming cannot be used, since the principle of optimality is not respected. We try therefore to detect the type of safe pruning that can be done.

We have proved that if at a frame a we have two paths P'_a and P''_a with $S''_a < S'_a$ and $L'_a < L''_a$, then at no frame $c \geq a$ will a path P''_c be forsaken for a path P'_c if $P'_a \subset P'_c$, $P''_a \subset P''_c$ and $P'_c \setminus P'_a \equiv P''_c \setminus P''_a$. We will note the order relation as $P''_a \prec P'_a$. We have further shown that a path P' may be discarded only for a lower cost one, P".

$$P' \prec P'' \Rightarrow C'_k < C''_k \tag{8}$$

Thus, algorithm 1 computes $S(M, X)$ and Q^* from equation (3).

By ordering the set of paths, according to Equation 8, we only need to check the line 1.2 of algorithm 1 up to the eventual insertion place. The last paths are candidates for pruning in line 1.1. In order for the pruning to be acceptable, we will prune only paths that were too long on the last state. An additional counter is needed for storing the state length. This counter is reset when the state is changed and is incremented at each advance with a frame.

```
procedure OneStep(W,X)
    SetOfPaths(1..N, 1..K)←∅
    for all frame=1; frame <= N; frame++ do
        for all state=1; state <= K; state++ do
            for all candidate $p_i$∈SetOfPaths(frame-1, 1..K) do
                Add($p_i$, SetOfPaths[frame, state])
            end
        end
    end
    SetOfPaths[frame, K] ← best of the candidates
end.
procedure Add(path, set-of-paths)
    for all $p_i$∈set-of-paths do
1.1     if path≺$p_i$ then
            delete $p_i$
        end
1.2     if $p_i$≺path then
            return
        end
    end
    Insert $p_i$ in set-of-paths
end.
```

Algorithm 1: *One Step Algorithm*

3.4 One pass confidence-based keyword spotting

3.4.1 The Method of Double Normalization

The corresponding confidence measure is defined as:

$$\frac{1}{NVP}\sum_{p_i\in VP}\frac{\sum_{pst\in p_i}-\log(pst)}{length(p_i)} \tag{9}$$

where NVP stands for the *number of visited phonemes* and VP stands for the *set of visited phonemes*. An average is computed over all posteriors *pst* of the emission probabilities for the time frames matched to the visited phoneme p_i. The function *length*(p_i) gives the number of time frames matched against p_i.

This method consists into a breath first Beam Search algorithm. It refers to a set of reduction rules and certain normalizations:

For the state q_G, in this method, the logarithm of the emission posterior is equal with zero. For each frame e and for each state s, the set of paths/probabilities of having the frame e in the state s is computed as the first N maxima (N can be finite) of the confidence measure for all paths in HMM $\overline{M}$ of length e and ending in the state s. The paths that according to the reduction rules will loose the final race when compared with another already known path, will be deleted as well.

We note a_1, p_1, l_1, a_2, p_2 and l_2 the confidence measure for the previous phonemes, the posterior in the current phoneme and the length in the current phoneme for the path Q_1, respectively the path Q_2. The rules that may be used for the reduction of the search space by discarding a path Q_1 for a path Q_2 are in this case any of the next ones:

1. $l_2 \geq l_1$, $A > 0$, $B \leq 0$ and $L_c^2 A + L_c B + C \geq 0$
2. $l_2 \geq l_1$, $A \geq 0$, $B \geq 0$ and $C \geq 0$
3. $l_2 \geq l_1$, $A \leq 0$, $C \geq 0$ and $L^2 A + LB + C \geq 0$
4. $l_2 \geq l_1$, $A = 0$, $B < 0$ and $LB + C \geq 0$

where $A = a_1 - a_2$, $B = (a_1 - a_2)(l_1 + l_2) + p_1 - p_2$, $C = (a_1 - a_2) l_1 l_2 + p_1 l_2 - p_2 l_1$, $L = L_{max} - \max\{l_1, l_2\}$, $L_c = -B/2A \geq 0$ and L_{max} is the maximum acceptable length for a phoneme.

By discarding paths only if one of the above rules is satisfied, the optimum defined by the confidence measure with double normalization can be guaranteed, if no phone may be avoided by the HMM M. Any HMM may be decomposed in HMMs with this quality. The 4-th rule is included in the 3-rd and its test is useless if the last one was already checked.

First test, $l_2 \geq l_1$ tells us if Q_2 has chances to eliminate Q_1, otherwise we will check if Q_1 eliminates Q_2. These tests were inferred from the conditions of maintaining the final maximal confidence measure while reduction takes place. In order to use the method of double normalization without decomposing HMMs that skip some phonemes, the previous rules are modified taking into account the number of visited phonemes for any path F_1 respectively F_2 and the number of phonemes that may follow the current state.

A simplified test may be:

- $l_2 \geq l_1$, $A \geq 0$, $p_1 \geq p_2$ respectively $F_2 \geq F_1$ for the HMMs that skips phonemes.

This test is weaker than the 2nd reduction rule. For example a path is eliminated by a second path if the first one has an inferior confidence measure (higher in value) for the the previous phonemes, a shorter length and the minus of the logarithm of the cumulated posterior in the current phoneme also inferior (higher in value) to that of the second one.

An additional confidence measure based on the maximal length, L_{max}, and on the maximum of the minus of the logarithm of the cumulated and normalized posterior in phoneme, P_{max}, can be used in order to limit the number of stored paths.

- $p > L_{max}P_{max}$ in any state
- $\frac{p}{l} > P_{max}$ at the output from a phoneme

where p and l are the values in the current phoneme for the minus of the logarithm of cumulated posterior and for the length of the path that is discarded. These tests allow for the elimination of the paths that are too long without being outstanding, respectively of the paths with phonemes having unacceptable scores, otherwise compensated by very good scores in other phonemes.

If N is chosen equal with one, the aforementioned rules are no longer needed, but always we propagate the path with the maximal current estimation of the confidence measure. The obtained results are very good, even if the defined optimum is guaranteed for this method only when N is bigger than the length of the sequence allowed by L_{max} or of the tested sequence.

The same approach is valid for the simple normalization, where the HMM for the searched word will be grouped into a single phoneme.

3.4.2 The Method of Real Fitting

We have also defined a new confidence measured that represents differently the exigencies of the recognition. Since the phonemes and the absent states can be modeled by the used HMMs, we find it interesting to request the fitting of each phoneme in the model with a section of the sequence. Therefore, we measure the confidence level of a subsequence as being equal with the maximum over all phonemes of the minus of the logarithm of the cumulated posterior of the phone, normalized with its length.

$$\max_{phonem \in Visited\ Phonems} \frac{\sum_{phonem} - \log(posteriors)}{phonem\ length} \tag{10}$$

The rule that may be used in this framework for the reduction of the number of visited paths is:

- Q_2 is discarded in favor of another path Q_1 if the confidence measure of the Real Fitting for the previous phonemes is inferior (higher in value) for Q_2 compared with Q_1, and if $p_1 \leq p_2$ and $l_2 \leq l_1$.

where p_1, l_1, p_2, l_2 represent the minus of the logarithm of the cumulated posterior respectively the number of frames in the current phoneme for the path Q_1 respectively Q_2.

Similarly to the previous method, the set of visited paths can be pruned by discarding those that:

- $p > L_{max}P_{max}$ in any state
- $\frac{p}{l} > P_{max}$ at the output from a phoneme

where p and l are the values in the current phoneme for the minus of the logarithm of the cumulated posterior and for the length of the path that is discarded. We recall that the meaning of the constants are the maximal length L_{max}, respectively the accepted maxima of the minus of the logarithm of the cumulated and normalized posterior in phoneme, P_{max}.

3.5 Conclusions

We have thus proposed a new method for keyword spotting, based on recent advances in confidence measures, using local posterior probabilities, but without requiring the explicit use of filler models.

A new algorithm, referred to as *Iterating Viterbi Decoding (IVD)*, to solve the above optimization problem with a simple DP process (not requiring to store pointers and scores for all possible ending and start times), at the cost of a few iterations. Other three beam-search algorithms corresponding to three different confidence measures were also described.

While the proposed approach allows for an easy generalization to more complex criteria, preliminary results obtained on the basis of 100 keywords (and without any specific tuning) appear to be particularly competitive to other alternative approaches.

3.6 The object of the invention consists of:

- Method of recognition of a subsequence using a direct maximization of confidence measures.

- The method of IVD for directly maximizing the confidence measures based on simple normalization.
- The use of the confidence measure and method of recognition named 'Real Fitting', based on individual fitting for each phoneme.
- Methods of recognition using simple and double normalization by:
- combining these measures with additional confidence measures mentioned here, respectively the maximal length and real matching limitation.
- The use of the aforementioned methods in keyword recognition.
- The use of the aforementioned methods in subsequence recognition of organic matter.
- The use of the aforementioned methods in recognition of objects in images.

4 Best Mode for Carrying Out the Invention

Execution: It is necessary to use a computer, but the method can also be implemented in hardware.

1. A representation under the form of an HMM is obtained for the subsequences that are looked for (word, protein profile, section of an image of the object).
2. A tool will be obtained (eventually trained Ex: for speech recognition) for the estimation of the posteriors. For example multi-Gaussians, neuronal networks, clusters, database with Generalized Profiles and mutation matrices (PAM, BLOSSUM, etc.).
3. One of the proposed algorithms should be implemented. They yield close performance but the method of Real Fitting coupled with a well checked dictionary should perform best.

   For the first algorithm (IVD)

   (a) The classic algorithm of Viterbi is implemented with the modification that, for each pair $P = \langle sample, state \rangle$ one propagates the moments of transition between the state q_G and the states of the HMM M for the path that arrives at P. These are inherited from the path that wins the entrance in the pair P, excepting for the moment when their decision is taken, namely when they receive the index of the corresponding sample.

(b) $w = -\log P(M|X_b^e)$ is computed by subtracting from the cumulated posterior that is returned by the Viterbi algorithm for the path Q_b^e, the value $(N-(e-b+1)) * \varepsilon$ corresponding to the contribution of the states q_G and dividing the result through $e-b+1$. $e-b+1$ from the previous formula can be factorized outside the fraction.

(c) The initialization of ε is made with an expected mean value. One can use the w that is computed when the state q_G is associated with an emission posterior equal to the average of the best K emission probabilities of the current sample as done in the well-known "garbage on-line model". In this case, K is trained using the corresponding technique.

The next 'Beam search' algorithms, are implemented according to the description in the corresponding sections. For each pair $P = \langle sample, state \rangle$ one computes for each corresponding path the sum and length in the last phoneme, as well as the sum over the normalized cumulated posteriors of the previous phonemes (and their number). Also, the entrance and exit samples into the HMM M are computed and propagated like in the previous method, in order to ensure the localization of the subsequence.

4. If one searched entity (keyword, sequence, object) can have several HMM models, all of them are taken into consideration as competitors. This is the case of the words with several pronunciations (or of the objects that have different structures in different states, for the recognition in images).

   After the computation of the confidence measure for each model of the subsequences, one eliminates those with a confidence measure in disagreement with a 'threshold' that is trained for the configuration and the goal of the given application. For example, for speech recognition with neuronal networks and minus of the logarithm of the posteriors, the 'threshold' is chosen in the wanted point of the ROC curve obtained in tests.

5. The remained alternatives are extracted in the order of their confidence measure and with the elimination of the conflicting alternatives until exhaustion. Each time when an alternative is eliminated, the searched entity with the corresponding HMM is re-estimated for the remaining sections in the sequence in which the search is performed.

If the new confidence measure passes the test of the 'threshold', then it will be inserted in the position corresponding to its score in the queue of alternatives.

6. The successful alternatives can undergo tests of superior levels like for example a question of confirmation for speech recognition, opinion of one operator, etc.

7. For objects recognition in images:

   Posteriors are obtained by computing a distance between the color of the model and that of element in the section of the image. If the context requires, the image will be preprocessed to ensure a certain normalization (Ex: changeable conditions of light will make necessary a transformation based on the histogram).

   The phonemes of the speech recognition correspond to parts of the object. The structure (existence of transitions and their probabilities) can be modified, function of the characteristics detected along the current path. For example, after detecting regions of the object with certain lengths, one can estimate the expected length of the remaining regions. Thus, the number of the expected samples for the future states can be established and the HMM attached to the object will be configured accordingly.

   A direction is scanned for the detection of the best fitting and afterwards, other directions will be scanned for discovering new fittings, as well as for testing the previous ones. The final test will be certified by classical methods such as cross-correlation or by the analysis of the contours in the hypothesized position.

5 Industrial Applicability

Here we present some examples for the application of the proposed method in the industry:

- The recognition of keywords begins to be used in answering automates of banking system as well as telephone and automates for control, sales or information. The method offers a possibility to recognize keywords in spontaneous speech with multiple speakers.

- The recognition of DNA sequences is important for the study of the human Genome. One of the biggest problem of the involved techniques consists in the high quantity of data that have to be processed.

- The recognition of objects in images is used, among others, in cartography and in the coordination of industrial robots. The method allows a quick estimation of the position of the objects in scenes and can be validated with extra tests, using classical methods of cross-correlation.

References

[1] Bernardis, G. and Bourlard, H., "Improving posterior-based confidence measures in hybrid HMM/ANN speech recognition systems," *Proceedings of Intl. Conf. on Spoken Language Processing* (Sydney, Australia), pp. 775-778, 1998.

[2] Bourlard, H. and Morgan, N., *Connectionist Speech Recognition - A Hybrid Approach*, Kluwer Academic Publishers, 1994.

[3] Bourlard, H., D'Hoore, B., and Boite, J.-M., "Optimizing recognition and rejection performance in wordspotting systems," *Proc. of IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing* (Adelaide, Australia), pp. I:373-376, 1994.

[4] Bridle, J.S., "An efficient elastic-template method for detecting given words in running speech," *Proc. of the Brit. Acoust. Soc. Meeting*, pp. 1-4, April 1973.

[5] Rohlicek, J.R., "Word spotting," in *Modern Methods of Speech Processing*, R.P. Ramachandran and R. Mammone (Eds.), Kluwer Academics Publishers, pp. 123-157, 1995.

[6] Rose, R.C. and Paul, D.B., "A hidden Markov model based keyword recognition system," *Proc. of ICASSP'90*, pp. 129-132, 1990.

[7] Sukkar, R.A. and Lee, C.-H., "Vocabulary independent discriminative utterance verification for nonkeyword rejection in subword based speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 4, no. 6, pp. 420-429, 1996.

[8] Williams, G. and Renals, S., "Confidence measures for hybrid HMM/ANN speech recognition," *Proceedings of Eurospeech'97*, pp. 1955-1958, 1997.

[9] Wilpon, J.G., Rabiner, L.R., Lee C.-H., and Goldman, E.R., "Application of hidden Markov models of keywords in unconstrained speech," *Proc. of ICASSP'89*, pp. 254-257, 1989.